APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/823,182	03/29/2001	Daniel M. Lavery	42390.P10809	9287

7590   01/13/2005

R. Alan Burnett
BLAKELY, SOKOLOFF, TAYLOR & ZARMAN LLP
Seventh Floor
12400 Wilshire Boulevard
Los Angeles, CA 90025-1026

EXAMINER
FOWLKES, ANDRE R

ART UNIT	PAPER NUMBER
2122	

DATE MAILED: 01/13/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

	Application No.	Applicant(s)
Office Action Summary	09/823,182	LAVERY ET AL.
	Examiner	Art Unit	
	Andre R. Fowlkes	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE <u>3</u> MONTH(S) FROM
THE MAILING DATE OF THIS COMMUNICATION.
- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed
  after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
  Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any
  earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *17 September 2004*.

2a)☒ This action is **FINAL**.　　2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is
closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-30* is/are pending in the application.

　　4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-30* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *04 June 2001* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

　　Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

　　Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

　　a)☐ All　b)☐ Some * c)☐ None of:

　　　1.☐ Certified copies of the priority documents have been received.

　　　2.☐ Certified copies of the priority documents have been received in Application No. _____.

　　　3.☐ Copies of the certified copies of the priority documents have been received in this National Stage
application from the International Bureau (PCT Rule 17.2(a)).

　　* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1)☐ Notice of References Cited (PTO-892)
2)☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3)☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
　　Paper No(s)/Mail Date _____.

4)☐ Interview Summary (PTO-413)
　　Paper No(s)/Mail Date. _____ .
5)☐ Notice of Informal Patent Application (PTO-152)
6)☐ Other: _____.

DETAILED ACTION

1.   This action is in response to the amendment, filed 9/7/04.

Claim Rejections - 35 USC § 102

2.   The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that

form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in a patent granted on an application for patent by another filed in the United States before the invention thereof by the applicant for patent, or on an international application by another who has fulfilled the requirements of paragraphs (1), (2), and (4) of section 371(c) of this title before the invention thereof by the applicant for patent.

The changes made to 35 U.S.C. 102(e) by the American Inventors Protection Act

of 1999 (AIPA) and the Intellectual Property and High Technology Technical

Amendments Act of 2002 do not apply when the reference is a U.S. patent resulting

directly or indirectly from an international application filed before November 29, 2000.

Therefore, the prior art date of the reference is determined under 35 U.S.C. 102(e) prior

to the amendment by the AIPA (pre-AIPA 35 U.S.C. 102(e)).

3.   Claims 1-3, 6, 9-12, 14, 16-18, 20-22, 23-25, and 27-29 are rejected under 35

U.S.C. 102(e) as being anticipated by Dehnert et al. (Dehnert), U.S. Patent No.

6,059,839.

As per claim 1, Dehnert discloses a method for performing memory

disambiguation in a compiler, comprising:

- **determining memory objects corresponding to memory references in one or more source files being compiled** (col. 3 lines 60, "determining address data of memory references in software code"),

- **creating a memory disambiguation** file **for each of the memory references, each memory disambiguation file identifying information particular to the memory reference it is associated with so as to preserve high-level and intermediate-level semantic information** (Fig. 2, the Intermediate Representation File (IRF) contains information particular to the memory reference it is associated with, as disclosed in steps 202, 203, 206 and 207, and associated text, such as (col. 5 line 11 – col. 6 line 49), "The inter procedural analysis phase (IPA) analyzes the information propagated by the IRF, step 206. IPA analyses the variables globally across all procedures... It pulls all variable data from the IRF. The IPA determines which variables must be treated as address-saved, inhibiting optimization as described above. The output of the IPA phase is precise address-taken information about all global variables and formal parameter variables. In the global optimization phase, the back end of the compiler uses the precise address-taken information from the summary tables generated by IPL and IPA (i.e. from the IRF) to optimize the code as thoroughly as possible, in step 207. The precise address-taken information generated by the IPL phase and the IPA phase of the present invention have provided for the full disambiguation of the memory references (variables, parameters, and the like). The references which cannot be optimized without inducing errors are particularly identified").

- **creating a symbolic memory reference** file **associated with each memory
disambiguation** file, **including information on whether the memory reference is
indirect or direct and access to symbol table information for a pointer to the
memory object for indirect references or the memory object for direct references**
(Fig. 2, the IRF contains information, including whether the memory reference is indirect
or direct and access to the information contained in a symbol table, as disclosed in item
205, and associated text, (e.g. col. 5 line 11 – col. 6 line 49)"),

- **determining if potentially dependent memory references are dependent or
independent based on information contained in the disambiguation** files **for those
memory references and their associated symbolic memory reference** files (col. 4
lines 1-3, "The compiler analyzes the operations the program actually performs to
determine which (memory references are independent and) can be optimized").

As per claim 2, the rejection of claim 1 is incorporated and further, Dehnert
discloses **determining if the memory references are redundant based on
information contained in the disambiguation** files **for those memory references
and their associated symbolic memory reference** files (col. 2 lines 2-4, "the compiler
may (use the information in the IRF (i.e. the memory disambiguation file) to) ... place
code in locations such that a particular calculation called for in the code can be
performed once, instead of (many times)").

As per claim 3, the rejection of claim 1 is incorporated and further, Dehnert

discloses **determining a relative difference in starting addresses for two memory**

references (col. 2 lines 15, "The compiler must make sure not to move certain units of

code, especially memory references, past other units of code which refer to the same

memory location", and the relative difference between starting addresses for two

memory locations must be determined before the compiler moves code units).

As per claim 6, the rejection of claim 1 is incorporated and further, Dehnert

discloses that **the disambiguation** file **associated with the memory object includes**

access capability **that is used to access data dependence information** (col. 4 lines 1-

3, "The compiler analyzes the operations the program actually performs to determine

(data dependence information and decide which memory references) can be

optimized").

As per claim 9, the rejection of claim 1 is incorporated and further, Dehnert

discloses **using information identified by disambiguation** files **to determine sets of**

local memory objects that are not referenced after they are modified (col. 4 lines 8-

11, "(The system) determines what kind of references occur for each variable ... (the

system, then) summarizes all these local cases").

As per claim 10, the rejection of claim 1 is incorporated and further, Dehnert

discloses **determining if two memory references access overlapping memory**

locations based on information contained in the disambiguation files **for those**

memory references and their associated symbolic memory reference files (col. 5

lines 37-40, "The compiler of the present invention determines the nature of the

operations the program actually performs with the addresses. The ultimate objective is

determining which of the (memory references access overlapping memory locations).").

As per claim 11, the rejection of claim 10 is incorporated and further, Dehnert

discloses **determining particularities of an overlap between two overlapping**

memory references (col. 5 lines 56-59, "(The system) analyzes all of the procedures

and all of the data references inside the procedures)...(The system, then) compares

and categorizes all local references to the variables").

As per claim 12, the rejection of claim 1 is incorporated and further, Dehnert

discloses:

- **determining the functions executed corresponding to function calls in the**

one or more source files being compiled (Fig 2 item 204, "Analyze local procedures

and data calls", and associated text, (e.g. col. 5 line 11 – col. 6 line 49)),

-creating a disambiguation file **for each function call, each disambiguation**

file **identifying information particular to the function call it is associated with so as**

to preserve high-level and intermediate level semantic information (Fig 3 items

307, "Analyze attributes for all procedure calls", and 315, "Summarize procedure call

variable information (in a file)", and associated text, (e.g. col. 6 line 50 – col. 8 line 12)),

-creating a symbolic function call file **associated with each disambiguation**

file, including information on whether the function call is indirect or direct and

access to symbol table information for the pointer or function respectively (Fig 3

items 308, "(determine if procedure calls are) direct or indirect", and 315, "Summarize

procedure call variable information (in a file)", and associated text, (e.g. col. 6 line 50 –

col. 8 line 12)),

- determining if potentially dependent calls and memory references are

dependent or independent for the function calls based on information contained

in the disambiguation files for the calls and memory references, their associated

symbolic file, an analysis of each function to determine the set of memory

locations modified or referenced by the function (col. 4 lines 1-3, "The compiler

analyzes the operations the program actually performs to determine which (memory

references are independent and) can be optimized").

As per claim 14, the rejection of claim 1 is incorporated and further, Dehnert

discloses **the disambiguation file for an indirect memory reference contains a link**

to a set of memory objects accessible via the pointer as determined by points-to

analysis (Fig. 5, item 502, and associated text (e.g. col. 8 lines 36-47) show the

summary of the disambiguation file for an indirect memory reference).

As per claims 16-18 and 20-22, this is a system version of the claimed method

discussed above, in claims 1- 3 and 9-12, wherein all claimed limitations have also been

addressed and/or cited as set forth above. For example, see Dehnert's apparatus and

method for compiler identification of address data (col. 2:2-8:47).

As per claims 23-25 and 27-29, this is an article of manufacture version of the

claimed method discussed above, in claims 1-3 and 9-12, wherein all claimed limitations

have also been addressed and/or cited as set forth above. For example, see Dehnert's

apparatus and method for compiler identification of address data (col. 2:2-8:47).

Claim Rejections - 35 USC § 103

4.   The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all

obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set
> forth in section 102 of this title, if the differences between the subject matter sought to be patented and
> the prior art are such that the subject matter as a whole would have been obvious at the time the
> invention was made to a person having ordinary skill in the art to which said subject matter pertains.
> Patentability shall not be negatived by the manner in which the invention was made.

5.   Claims 4, 5, 7, 13, 15 and 30 are rejected under 35 U.S.C. 103(a) as being

unpatentable over Dehnert et al. (Dehnert), U.S. Patent No. 6,059,839 in view of Novak

et al. (Novak), "A simple mechanism for improving the accuracy and efficiency of

instruction-level disambiguation".

As per claim 4, the rejection of claim 1 is incorporated and further, Dehnert

doesn't explicitly disclose that **the disambiguation token comprises a data structure**

including a plurality of links to data objects in which disambiguation information are stored.

However, Novak, in an analogous environment, discloses that **the disambiguation token comprises a data structure including a plurality of links to data objects in which disambiguation information are stored** (p. 293, ¶ 3 lines 1-4, "the data structure created by the compiler during parsing and source-level analysis—which maintain, for example, defuse and alias information (i.e. disambiguation info)", and the token is described on p. 293, ¶ 5 line 4 – p. 294, ¶ 1 line 2, "allowing the front-end to communicate higher level, memory reference information to the back-end by associating each reference with a portion of a hierarchical decomposition of the program's address space (token)").

Therefore, it would have been obvious to a person of ordinary skill in the art, at the time the invention was made, to incorporate the teachings of Novak into the system of Dehnert to have the disambiguation token comprise a data structure including links to data objects in which disambiguation information is stored. The modification would have been obvious because one of ordinary skill in the art would want to allow the front end of a compiler to communicate higher level memory reference information to the back-end in order to allow for increased efficiency and accuracy of optimization of the code (Novak, p. 293 ¶ 5 line 1 – p. 294 ¶ 1 line 2).

As per claim 5, the rejection of claim 4 is incorporated and further, Dehnert discloses that **the data structure is embedded in memory reference operators of an**

intermediate language produced during the compilation of the one or more

source files (col. 6 lines 20-21, "The IRF (i.e. intermediate language produced during

compilation) includes the additional data gathered about the variables").

As per claim 7, the rejection of claim 1 is incorporated and further, Dehnert

doesn't explicitly disclose that **the disambiguation token contains a link to address**

base and offset information for the memory reference that is used for low-level

disambiguation.

However, Novak, in an analogous environment, discloses that **the**

disambiguation token contains a link to address base and offset information for

the memory reference that is used for low-level disambiguation (p. 293, ¶ 5 line 4 –

p. 294, ¶ 1 line 2, "allowing the front-end to communicate higher level, memory

reference information to the back-end by associating each reference with a portion of a

hierarchical decomposition of the program's address space (i.e. address base and

offset information)").

Therefore, it would have been obvious to a person of ordinary skill in the art, at

the time the invention was made, to incorporate the teachings of Novak into the system

of Dehnert to have the disambiguation token contains a link to address base and offset

information for the memory reference that is used for low-level disambiguation. The

modification would have been obvious because one of ordinary skill in the art would

want to use as much information as possible in order to maximize the possible code

optimizations (Novak, p. 293 ¶ 5 line 1 – p. 294 ¶ 1 line 2).

As per claim 13, the rejection of claim 1 is incorporated and further, Dehnert

doesn't explicitly disclose that **the disambiguation token contains a link to type**

information associated with the memory reference.

However, Novak, in an analogous environment, discloses that **the**

disambiguation token contains a link to type information associated with the

memory reference (p. 297 ¶ 1 lines 3-7"(the disambiguation token contains type

information to allow the) dependence analysis tool (to ensure that) ... pointer variables

will be associated only with paths for which the appropriate 'type' node is present").

Therefore, it would have been obvious to a person of ordinary skill in the art, at

the time the invention was made, to incorporate the teachings of Novak into the system

of Dehnert to have the disambiguation token contain a link to type information

associated with the memory reference. The modification would have been obvious

because one of ordinary skill in that art would have wanted use type information to

improve the disambiguation of pointer references (Novak, p. 297 ¶ 1 lines 7-9).

As per claim 15, the rejection of claim 1 is incorporated and further, Dehnert

doesn't explicitly disclose **using the disambiguation token and the symbolic**

memory reference file as an interface or means of communication between

various software components of a disambiguator that performs memory

disambiguation functions and clients of the disambiguator.

However, Novak, in an analogous environment, discloses **using the
disambiguation token and the symbolic memory reference** file **as an interface or
means of communication between various software components of a
disambiguator that performs memory disambiguation functions and clients of the
disambiguator** (p. 293, ¶ 5 line 4 – p. 294, ¶ 1 line 2, "allowing the front-end (software
component) to communicate higher level, memory reference information to the back-
end (software component) by associating each reference with a portion of a hierarchical
decomposition of the program's address space").

Therefore, it would have been obvious to a person of ordinary skill in the art, at
the time the invention was made, to incorporate the teachings of Novak into the system
of Dehnert to use the disambiguation token and the symbolic memory reference file as
an interface or means of communication between various software components of a
disambiguator that performs memory disambiguation functions and clients of the
disambiguator. The modification would have been obvious because one of ordinary skill
in the art would want to use as much information as possible in order to maximize the
possible code optimizations at each level of compilation (Novak, p. 293 ¶ 5 line 1 – p.
294 ¶ 1 line 2).

As per claim 30, Dehnert in combination with Novak also discloses such claimed
limitations as addressed in claim 15 above.

6.   Claims 8, 19, and 26 are rejected under 35 U.S.C. 103(a) as being unpatentable

over Dehnert et al. (Dehnert), U.S. Patent No. 6,059,839 in view of Rountev et al.

(Rountev), "Off-line variable substitution for scaling points-to analysis".

        As per claim 8, the rejection of claim 1 is incorporated and further, Dehnert

doesn't explicitly disclose **substituting a direct memory reference for an indirect**

memory reference.

        However, Rountev, in an analogous environment, discloses **substituting a**

direct memory reference for an indirect memory reference (p. 2, col. L lines 30-33,

"we propose off-line variable substitution").

        Therefore, it would have been obvious to a person of ordinary skill in the art, at

the time the invention was made, to incorporate the teachings of Rountev into the

system of Dehnert to have substitution of direct memory references for indirect memory

references. The modification would have been obvious because one of ordinary skill in

the art would want to reduce the cost of points-to analysis (p. 2, col. L lines 30-33).

        As per claims 19 and 26, Dehnert in combination with Rountev also discloses

such claimed limitations as addressed in claim 8 above.

                              Response to Arguments

7.   Applicants arguments have been considered but they are not persuasive.

In the remarks, the applicant has argued substantially that:

1)   Neither Dehnert, alone or in combination with Novak teach creating a memory

disambiguation token for each of the memory references, each memory disambiguation

token identifying information particular to the memory reference it is associated with so

as to preserve high-level and intermediate-level semantic information, at p. 10:10-11:21

and 11:27-12:7.

Examiner's response:

1)   The examiner disagrees with the applicants characterization of the applied art.

The examiner contends that the cited intermediate representation file (IRF) does contain

information particular to the memory references it is associated with so as to preserve

high-level and intermediate level semantic information, including "all of the information

(either directly or through other links) necessary to perform memory disambiguation" as

required by independent claims 1, 16 and 23 and stated by applicant at p. 11:7-11 of the

amendment. Dehnert describes how the IRF contains directly or indirectly contains all

of the information needed to perform memory disambiguation in the following, col. 6:26-

49, "The inter procedural analysis phase (IPA) analyzes the information propagated by

the IRF, step 206. IPA analyses the variables globally across all procedures... It pulls all

variable data from the IRF. The IPA determines which variables must be treated as

address-saved, inhibiting optimization as described above. The output of the IPA phase

is precise address-taken information about all global variables and formal parameter

variables. In the global optimization phase, the back end of the compiler uses the

precise address-taken information from the summary tables generated by IPL and IPA

(and the IRF) to optimize the code as thoroughly as possible, in step 207. The precise address-taken information generated by the IPL phase and the IPA phase of the present invention have provided for the full disambiguation of the memory references (variables, parameters, and the like). The references which cannot be optimized without inducing errors are particularly identified. The compiler can then optimize other references, for example, by moving them outside of loops. In doing so, the code is enhanced in speed and efficiency. The output of this step is fully optimized, machine executable code, in step 208"

Conclusion

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Andre R. Fowlkes whose telephone number is (571)

272-3697. The examiner can normally be reached on Monday - Friday, 8:00am-

4:30pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's

supervisor, Tuan Q. Dam can be reached on (571)272-3695. The fax phone number for

the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the

Patent Application Information Retrieval (PAIR) system. Status information for

published applications may be obtained from either Private PAIR or Public PAIR.

Status information for unpublished applications is available through Private PAIR only.

For more information about the PAIR system, see http://pair-direct.uspto.gov. Should

you have questions on access to the Private PAIR system, contact the Electronic

Business Center (EBC) at 866-217-9197 (toll-free).

ARF

TUAN DAM
SUPERVISORY PATENT EXAMINER